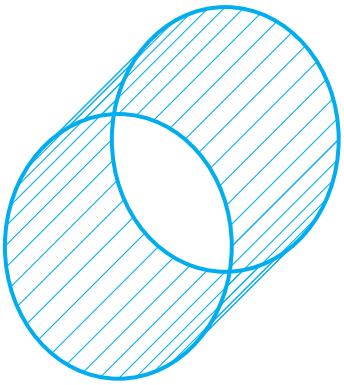
ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA TOÁN - TIN HỌC



Khoa Toán - Tin học
Fac. of Math. & Computer Science

KHÓA LUẬN TỐT NGHIỆP

PHÂN TÍCH CẢM XÚC DỰA TRÊN KHÍA CẠNH

CHUYÊN NGÀNH PHƯƠNG PHÁP TOÁN TRONG TIN

TP Hồ Chí Minh, 07.2022

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA TOÁN - TIN HỌC

KHÓA LUẬN TỐT NGHIỆP

PHÂN TÍCH CẢM XÚC DỰA TRÊN KHÍA CẠNH

CHUYÊN NGÀNH PHƯƠNG PHÁP TOÁN TRONG TIN

Người thực hiện: NGUYỄN QUỐC BẢO
Mã số sinh viên: 18110053
Giảng viên hướng dẫn: TS. NGÔ MINH MẪN

TP Hồ Chí Minh, 07.2022

Lời cảm ơn

Lời đầu tiên, tôi xin cảm ơn tất cả quý Thầy Cô giảng viên đã dành thời gian đánh giá luận văn tốt nghiệp của tôi, đặc biệt tôi muốn bày tỏ lòng biết ơn sâu sắc đến **TS. Ngô Minh Mẫn**, người đã trực tiếp giảng dạy và hướng dẫn tôi thực hiện luận văn này. Nhờ sự giúp đỡ nhiệt tình của thầy, bản thân tôi học hỏi được rất nhiều khi có cơ hội làm việc cùng thầy.

Bên cạnh đó, tôi xin chân thành cảm ơn các thầy, cô ở khoa Toán – Tin học, đặc biệt là **TS. Nguyễn Thanh Bình** - thầy trưởng bộ môn Phương pháp toán trong tin, đã truyền cảm hứng cũng như tạo điều kiện thuận lợi cho tôi hoàn thành luận văn tốt nghiệp.

Tôi cũng xin gửi lời cảm ơn đến các bạn trong lớp đại học chính quy tài năng khóa 18 và chị **Phạm Phi Nhung** đã cùng đồng hành và hỗ trợ tôi trong quãng thời gian thực hiện khóa luận.

Trong quá trình thực hiện khoá luận khó tránh khỏi những sai sót, tôi rất mong nhận đóng góp ý kiến từ quý Thầy Cô và các bạn để khoá luận được hoàn thiện hơn.

Một lần nữa, xin chân thành cảm ơn tất cả.

Tp.Hồ Chí Minh, tháng 7 năm 2022

Tác giả

Nguyễn Quốc Bảo

Tóm tắt nội dung

Nhu cầu về mua sắm trực tuyến ngày càng gia tăng, dẫn đến nguồn dữ liệu phản hồi/ý kiến của khách hàng về các sản phẩm và dịch vụ ngày càng nhiều. Khai thác về dữ liệu này nhằm cải thiện chất lượng sản phẩm, dịch vụ và tăng uy tín cho nhà cung cấp hơn, thu được lợi nhuận tốt nhất.

Đề tài này chúng tôi nghiên cứu bài toán phân tích cảm xúc dựa trên các khía cạnh trong các phản hồi/ý kiến của khách hàng trên tập dữ liệu thu thập từ các trang thương mại điện tử.

Trong luận văn đề xuất một số phương pháp trích xuất các khía cạnh có trong bình luận của khách hàng. Bên cạnh đó, từ các khía cạnh đã trích xuất sẵn thực hiện bước phân loại cảm xúc trên các khía cạnh đó. Nói cách khác, thay vì tạo ra hai mô hình riêng biệt thực hiện 2 nhiệm vụ riêng biệt, luận văn hướng đến một mô hình đa nhiệm vụ được trình bày qua từng chương như sau.

Bố cục luận văn

- **Chương 1:** Giới thiệu tổng quan về đề tài liên quan tới lĩnh vực xử lý ngôn ngữ tự nhiên, đồng thời giới thiệu các hướng đã tiếp cận và đề xuất hướng giải quyết cho đề tài.
- **Chương 2:** Sơ nét về các kiến thức cơ sở về xử lý ngôn ngữ tự nhiên và nhiệm vụ phân loại.
- **Chương 3:** Trình bày hướng tiếp cận và các mô hình cho đề tài.
- **Chương 4:** Mô tả và phân tích cấu trúc dữ liệu, hình thức gán nhãn cho dữ liệu và các cách thức xử lý dữ liệu, cách thức triển khai và kết quả thực nghiệm.
- **Chương 5:** Tổng kết và định hướng phát triển.

Mục lục

Danh mục ý nghĩa các ký hiệu

Từ khóa	ý nghĩa
NLP	natural language processing (Xử lý ngôn ngữ tự nhiên)
AI	artificial intelligence (trí tuệ nhân tạo)
ABSA	aspect-based sentiment anlysis (phân tích tình cảm dựa trên khía cạnh)
AE	Aspect Extraction (trích xuất khía cạnh)
APC	Aspect Polarity Classification (phân loại phân cực khía cạnh)
NER	Named Entity Recognition (nhận dạng thực thể được đặt tên)
positive, negative, and neutral	tích cực, tiêu cực và bình thường.
word embedding	nhúng từ
token	mã biểu diễn
tokenization	mã hóa
RNN	Recurrent Neural Network (Mạng nơ ron truy hồi)
Encoder & Decoder	bộ mã hóa & bộ giải mã
seq2seq	sequence to sequence (trình tự)
param	tham số
Transformer	biến đổi

Danh sách hình vẽ

Chương 1

Tổng quan

1.1 Giới thiệu bài toán

Với sự tiến bộ của công nghệ, hầu hết tất cả các công ty đều xây dựng hệ thống tự động lấy ý kiến khách hàng nhằm cải thiện dịch vụ. Các thông tin từ hệ thống này được lưu trữ lại theo từng khoảng thời gian tạo thành nguồn dữ liệu. Từ đó, để hiểu được những phản hồi của khách hàng, mỗi công ty đều có phòng ban thực hiện phân tích trạng thái cảm xúc của khách hàng, xu hướng của khách hàng. Tuy nhiên khi lượng phản hồi ngày càng tăng, việc phân loại cần bổ sung quá nhiều người làm ảnh hưởng đến chi phí hoạt động vì vậy một công cụ giúp phân loại được trạng thái cảm xúc của khách hàng trở nên cần thiết hơn.

Mỗi công ty có các thang đo để đánh giá, phân loại phản hồi của người dùng khác nhau, phổ biến nhất là hình thức đánh giá dạng thang đo theo các mức điểm rating từ 1 đến 5 sao để cho thấy cảm xúc tổng quát của khách hàng đối với sản phẩm hoặc dịch vụ. Tuy nhiên, ở phương pháp đánh giá này, việc phân tích tình cảm tiêu chuẩn (standard sentiment) chỉ đề cập đến việc phân loại tình cảm tổng thể của một câu hoặc một văn bản, không bao gồm các thông tin quan trọng khác như chủ đề, khía cạnh hoặc bất kỳ từ khóa chính trong câu. Từ vấn đề đó, ABSA trở thành một loại nhiệm vụ mới được quan tâm nhiều hơn trong dạng bài toán xử lý ngôn ngữ tự nhiên.

ABSA được chia làm 2 phần khác nhau gồm Aspect Extraction và Aspect Polarity
Classification, trong đó:

- Aspect Extraction (AE) là nhiệm vụ trích xuất các khía cạnh/ chủ đề/ đối tượng (aspect) cụ thể trong văn bản.
- Aspect Polarity Classification (APC) với nhiệm vụ phân loại các trạng thái cảm xúc (positive, negative, neutral) dựa trên từng aspect đã được xác định trong câu.

Một ví dụ cụ thể sự khác biệt giữa cách đánh giá truyền thống và đánh giá theo khía cạnh:

"Sản phẩm này rất hữu ích, giao hàng rất nhanh nhưng đóng gói chưa kỹ" và rating: 4/5 sao

Theo bài toán phân loại tình cảm truyền thống thì rating 4 sao được tính "positive".

Theo bài toán ABSA, đầu ra là **"Sản phẩm"**, **"giao hàng"** và **"đóng gói"**, đi cùng với kết quả phân loại tình cảm lần lượt là **"positive**, **"positive"**, **"negative"**. Nói cách khác thì kết quả sẽ được hiểu khách hàng đánh giá "positive" cho "sản phẩm", "positive" cho "giao hàng" và "negative" cho phần "đóng gói".

1.2 Các hướng tiếp cận trước đó

Nhiều nhà nghiên cứu đã xây dựng các mô hình tương ứng với từng nhiệp vụ AE và APC. Trong đó:

- ATAE-LSTM Wang et al. (2016)[19] là một mạng dựa trên LSTM cổ điển cho nhiệm vụ APC, áp dụng cơ chế chú ý để tập trung vào các từ quan trọng trong ngữ cảnh. Bên cạnh đó, ATAE-LSTM bổ sung tính năng nhúng khía cạnh và các tính năng đã học để sử dụng đầy đủ các tính năng khía cạnh. ATAE-LSTM có thể được điều chỉnh cho phù hợp với bộ dữ liệu đánh giá của Trung Quốc.
- ATSM-S Peng et al. (2018)[20] là mô hình cơ sở của các biến thể ATSM cho nhiệm vụ ABSA cho ngôn ngữ Trung Quốc. Mô hình này học câu và thuật ngữ khía cạnh ở ba góc độ chi tiết.
- GANN là mô hình mạng nơ-ron mới cho nhiệm vụ APC nhằm giải quyết những thiếu sót của RNN và CNN truyền thống. GANN đã áp dụng Gate Truncation RNN (GTR) để tìm hiểu các đại diện đầu mối cảm tính phụ thuộc vào khía cạnh thông tin. GANN đã có được hiệu suất APC hiện đại nhất trên bộ dữ liệu đánh giá của Trung Quốc.
- AEN-BERT Song et al. (2019)[21] là một mạng mã hóa chuyên dụng dựa trên mô hình BERT được đào tạo trước, nhằm mục đích giải quyết việc phân loại phân cực khía cạnh.
- BERT-PT Xu, Liu, Shu and Yu (2019) là một mô hình được BERT điều chỉnh cho nhiệm vụ đọc hiểu đánh giá (RRC), một nhiệm vụ lấy cảm hứng từ đọc hiểu máy (MRC), nó có thể được điều chỉnh để phân loại tình cảm ở cấp độ khía cạnh.
- BERT-BASE Devlin et al. (2019)[4] là mô hình BERT cơ bản được đào tạo trước. Chúng tôi điều chỉnh nó với tính năng học tập đa tác vụ ABSA, trang bị khả năng tự động trích xuất các thuật ngữ khía cạnh và phân loại cực tính các khía cạnh giống như mô hình LCF-ATEPC.
- BERT-SPC Song et al. (2019)[21] là một mô hình BERT được đào tạo trước được thiết kế cho nhiệm vụ phân loại theo cặp câu và cải thiện nhiệm vụ phụ APC của mô hình LCF-ATEPC.

- BERT-ADA Rietzler et al. (2019)[22] là một mô hình dựa trên BERT thích ứng với miền được đề xuất cho nhiệm vụ APC, mô hình này đã hoàn thiện mô hình BERT-BASE trên kho ngữ liệu liên quan đến nhiệm vụ. Mô hình này có được độ chính xác hiện đại trên tập dữ liệu Máy tính xách tay.

Lượng lớn các bài nghiên cứu về chủ đề này được các nhà nghiên cứu thực hiện.

Chương 2

Cơ sở lý thuyết

2.1 Định nghĩa chung

2.1.1 NLP và một số kỹ thuật trong NLP [2]

NLP là một phần trong những bài toán của AI với nhiệm vụ hướng đến khả năng làm cho máy tính có thể hiểu các ngôn ngữ tự nhiên của con người chẳng hạn như chữ viết, giọng nói, âm thanh, văn bản,... Trong thực tế, NLP là một dạng dữ liệu không cấu trúc (unstructure data) và cần một số kỹ thuật cao hơn để xử lý, trong đó hai kỹ thuật được sử dụng chính là phân tích syntax (cú pháp) và semantic (ngữ nghĩa).

Phân tích Syntax (Cú pháp)

Cú pháp là sự sắp xếp các từ trong một câu để có ý nghĩa ngữ pháp. NLP sử dụng syntax để đánh giá ý nghĩa từ một ngôn ngữ dựa trên các quy tắc ngữ pháp. Các kỹ thuật syntax bao gồm:

- **Parsing**: Một phương pháp phân tích ngữ pháp của một câu. Ví dụ: *"Quyển sách được mở"* thi câu này được chia thành các phần như "quyển sách" = "danh từ", "mở" = "động từ". Đây là một trong những điều hữu ích cho các tâc vụ xử lý.
- **Word segmentation**: mô tả hành động lấy một chuỗi văn bản và tạo ra các dạng từ tương ứng. Ví dụ:*"Một người cầm quyển sách"* thì thuật toán phân tích được mỗi từ được phân chia bởi các khoảng trắng trong câu.
- **Sentence breaking**: Thuật toán có thể nhận ra khoảng thời gian tách các câu bằng cách sử dụng ngắt câu.

- **Morphological segmentation**: Một từ được chia thành các phần tử nhỏ hơn được gọi là "morphemes". Đây là một trong những phương pháp hiệu quả truong dịch thuật và nhận dạng giọng nói.
- **Stemming**: Stemming là kỹ thuật dùng để biến đổi 1 từ về dạng gốc (được gọi là stem hoặc root form) bằng cách cực kỳ đơn giản là loại bỏ 1 số ký tự nằm ở cuối từ mà nó nghĩ rằng là biến thể của từ.

Phân tích Semantic (Ngữ nghĩa)

Phân tích ngữ nghĩa liên quan đến việc sử dụng và ý nghĩa đằng sau các từ. Một số kỹ thuật chung như:

- **Word sense disambiguation - WSD**: Đây là phương pháp xác định ý nghĩa của một từ dựa trên ngữ cảnh.
- **Named entity recognition - NER**: Đây là một trong những kỹ thuật trích xuất thêm thông tin về một số văn bản bằng cách gắn nhãn các từ khác nhau thành các danh mục được xác định trước như: người, địa điểm, thời gian, email,...

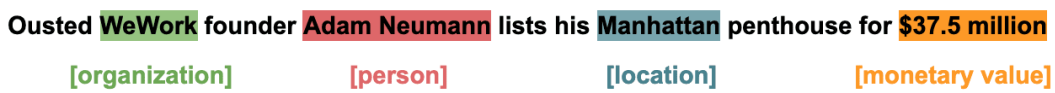


Figure 2.1: *Một ví dụ về NER [3]*

- **Natural language generation - NLG**: Đây là phương pháp được sử dụng trong một loạt các nhiệm vụ như Dịch máy, Chuyển giọng nói thành văn bản, chatbot, tự động sửa văn bản hoặc tự động hoàn thành văn bản.

2.1.2 IOB

IOB (viết tắt của inside, outside, beginning) là một định dạng gán nhãn cho các từ, cũng giống với NER dùng để xác định cấu trúc các danh mục (ví dụ: sản phẩm, mặt hàng, dịch vụ,...), với B là từ bắt đầu cho một danh mục, nối tiếp đó là I và O là nhãn không thuộc trong cấu trúc của các danh mục.

2.1.3 Aspect Extraction - AE [1]

Tương tự như nhiệm vụ NER, nhiệm vụ AE là một loại nhiệm vụ ghi nhãn trình tự. Cụ thể, trong nhiệm vụ AE, đầu vào của ví dụ xem xét "Giá cả không cao nhưng dịch vụ thì rất tốt." sẽ được chuẩn bị là $S = \{W1, W2, ...Wn\}$ và W là đại diện cho một token sau khi tokenization, $n = 10$ là tổng số các token.

Ví dụ câu trên sẽ được gắn nhãn theo dạng IOB, $Y = \{B\_asp, O, O, O, O, B\_asp, I\_asp, O, O, O\}$ thì kết quả của nhiệm vụ này mong muốn nhận được các nhãn "giá cả", "dịch vụ" là aspect.

2.1.4 Aspect polarity classification - APC [1]

APC là một nhiệm vụ phụ gồm nhiều cấp độ của phân tích giám sát, nhằm dự đoán phân cực cho các aspect đã được định sẵn. Giả sử rằng "Mặc dù dịch vụ chưa tốt nhưng giá cả hợp lý" là đầu vào cho nhiệm vụ APC, cũng gần giống với nhiệm vụ AE nhưng trong S có bao gồm cả aspect nối tiếp. Đầu ra mong đợi là "Giá" là postive, "dịch vụ" là negative.

2.2 Word2vec [5]

Word2vec là một mô hình đơn giản và nổi tiếng giúp tạo ra các biểu diễn embedding của từ trong một không gian có số chiều thấp hơn nhiều lần so với số từ trong từ điển. Ý tưởng của word2vec đã được sử dụng trong nhiều bài toán với dữ liệu khác xa với dữ liệu ngôn ngữ.

Ý tưởng cơ bản của word2vec

- Hai từ xuất hiện trong những văn cảnh giống nhau thường có ý nghĩa gần với nhau.
- Đoán được một từ nếu biết các từ xung quanh nó trong câu. Ví dụ, với câu "Hà Nội là ... của Việt Nam" thì từ trong dấu ba chấm khả năng cao là "thủ đô". Với câu hoàn chỉnh "Hà Nội là thủ đô của Việt Nam", mô hình word2vec sẽ xây dựng ra embeding của các từ sao cho xác suất để từ trong dấu ba chấm là "thủ đô" là cao nhất.

2.3 Attention

2.3.1 RNN [6]

Trong lý thuyết về ngôn ngữ, ngữ nghĩa của một câu được tạo thành từ mối liên kết của những từ trong câu theo một cấu trúc ngữ pháp. Nếu xét từng từ một đứng riêng lẻ ta không thể hiểu được nội dụng của toàn bộ câu, nhưng dựa trên những từ xung quanh ta có thể hiểu được trọn vẹn một câu nói. Như vậy cần phải có một kiến trúc đặc biệt hơn cho các mạng nơ ron biểu diễn ngôn ngữ nhằm mục đích liên kết các từ liền trước với các từ ở hiện tại để tạo ra mối liên hệ xâu chuỗi. Mạng nơ ron truy hồi đã được thiết kế đặc biệt để giải quyết yêu cầu này.

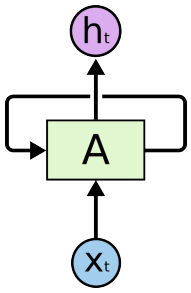


Figure 2.2: *Mạng nơ ron truy hồi với vòng lặp*

Hình trên biểu diễn kiến trúc của một mạng nơ ron truy hồi. Trong kiến trúc này mạng nơ ron sử dụng một đầu vào là một vector x_t và trả ra đầu ra là một giá trị ẩn h_t. Đầu vào được đấu với một thân mạng neutron A có tính chất truy hồi và thân này được đấu tới đầu ra h_t

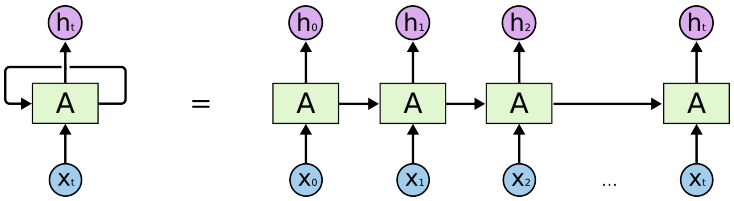


Figure 2.3: *Cấu trúc trải phẳng của RNN*

Hình trên biểu diễn kiến trúc trải thẳng của một RNN, với Vòng lặp A ở thân mạng nơ ron là điểm mấu chốt trong nguyên lý hoạt động của mạng nơ ron truy hồi. Đây là chuỗi sao chép nhiều lần của cùng một kiến trúc nhằm cho phép các thành phần có thể kết nối liền mạch với nhau theo mô hình chuỗi. Đầu ra của vòng lặp trước chính là đầu vào của vòng lặp sau.

Kiến trúc mạng nơ ron truy hồi RNN này tỏ ra khá thành công trong các tác vụ của deep learning như: Nhận diện dọng nói (speech recognition), các mô hình ngôn ngữ, mô hình dịch, chú thích hình ảnh (image captioning),.... Về mặt lý thuyết thì RNN có thể xử lý và lưu trữ thông tin của một chuỗi dữ liệu với độ dài bất kỳ. Tuy nhiên trong thực tế thì RNN chỉ tỏ ra hiệu quả với chuỗi dữ liệu có độ dài không quá lớn (short-term memory hay còn gọi là long-term dependency problem). Nguyên nhân của vấn đề này là do vanishing gradient problem (gradient được sử dụng để cập nhật giá trị của weight matrix trong RNN và nó có giá trị nhỏ dần theo từng layer khi thực hiện back propagation). Khi gradient trở nên rất nhỏ (có giá trị gần bằng 0) thì giá trị của weight matrix sẽ không được cập nhật thêm và do đó mạng Neuron sẽ dừng việc learning tại layer này. Đây cũng chính là lý do khiến cho RNN không thể lưu trữ thông tin của các timesteps đầu tiên trong một chuỗi dữ liệu có độ dài lớn.

2.3.2 Encoder và decoder [7]

Máy tính không thể học được từ các dữ liệu thô như bức ảnh, file text, file âm thanh, đoạn video. Do đó nó cần đến quá trình mã hóa thông tin sang dạng số và từ dạng số giải mã kết quả đầu ra. Đó chính là 2 quá trình encoder và decoder:

- Encoder: Là phrase chuyển input thành những features learning có khả năng học tập các task. Đối với model Neural Network, Encoder là các hidden layer. Đối với model CNN, Encoder là chuỗi các layers Convolutional + Maxpooling. Model RNN quá trình Encoder chính là các layers Embedding và Recurrent Neral Network.
- Decoder: Đầu ra của encoder chính là đầu vào của các Decoder. Phrase này nhằm mục đích tìm ra phân phối xác xuất từ các features learning ở Encoder từ đó xác định đâu là nhãn của đầu ra. Kết quả có thể là một nhãn đối với các model phân loại hoặc một chuỗi các nhãn theo tứ tự thời gian đối với model seq2seq.

2.3.3 Mô hình Sequence to Sequence [8]

Sequence to Sequence Model (Seq2seq) là một mô hình Deep Learning với mục đích tạo ra một output sequence từ một input sequence mà độ dài của 2 sequences này có thể khác nhau. Seq2seq được giới thiệu bởi nhóm nghiên cứu của Google vào năm 2014 trong bài báo **Sequence to Sequence with Neural Networks**. Mặc dù mục đích ban đầu của Model này là để áp dụng trong Machine Translation, tuy nhiên hiện nay Seq2seq cũng được áp dụng nhiều trong các hệ thống khác như Speech recognition, Text summarization, Image captioning,...

Seq2seq gồm 2 phần chính là Encoder và Decoder. Cả hai thành phần này đều được hình thành từ các mạng Neural Networks, trong đó Encoder có nhiệm vụ chuyển đổi dữ liệu đầu vào (input sequence) thành một representation với lower dimension còn Decoder có nhiệm vụ tạo ra output sequence từ representation của input sequence được tạo ra ở phần Encoder.

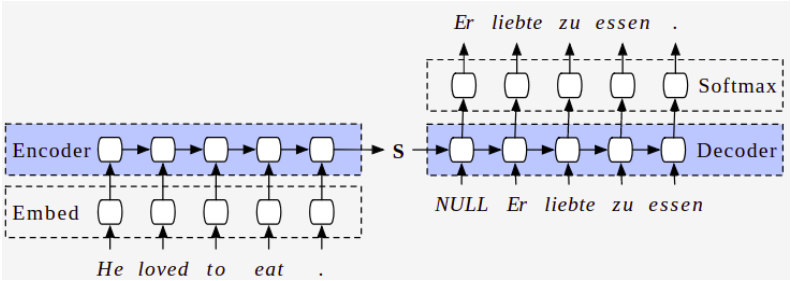


Figure 2.4: *Mô hình Sequence to Sequence trong dịch máy.*

2.3.4 Autoencoder [9]

Đây là phương pháp học cách biểu diễn dữ liêu hiệu quả thông qua học không giám sát sử dụng mạng nơ ron nhân tạo. Autoencoder bao gồm 3 phần chính:

- Encoder: Module có nhiệm vụ nén dữ liệu đầu vào thành một biểu diễn được mã hóa (coding), thường nhỏ hơn một vài bậc so với dữ liệu đầu vào
- Bottleneck: Module chứa các biểu diễn tri thức được nén (chính là output của Encoder), đây là phần quan trọng nhất của mạng bởi nó mang đặc trưng của đầu vào, có thể dùng để tái tạo ảnh, lấy đặc trưng của ảnh, ...
- Decoder: Module giúp mạng giải nén các biểu diễn tri thức và tái cấu trúc lại dữ liệu từ dạng mã hóa của nó, mô hình học dựa trên việc so sánh đầu ra của Decoder với đầu vào ban đầu (Input của Encoder)

2.3.5 Attention Mechanism [10]

Seq2seq sử dụng Encoder để tạo ra một vector representation lưu trữ thông tin của input sequence. Sau đó sử dụng Decoder để chuyển vector representation này thành output sequence. Với Machine Translation thì cả Encoder và Decoder đều được tạo thành từ RNN cell (LSTM hoặc GRU). Về mặt lý thuyết thì LSTM và GRU có thể lưu trữ thông tin của một sequence có độ dài lớn. Tuy nhiên, trong thực tế việc sử dụng một vector representation thường không thể lưu trữ được toàn bộ thông tin của input sequence. Nguyên tắc hoạt động chung của Attention Mechanism là tại mỗi Decoding Step, Decoder sẽ chỉ tập chung vào phần liên quan trong input sequence thay vì toàn bộ input sequence. Mức độ tập chung này được thiết lập bởi Attention weights như hình 2.5.

Như vậy, tại mỗi Decoding step, Decoder nhận 3 đầu vào là: Hidden state của decoding step trước, Output của step trước và Attention vector. Attention vector chứa Attention weight của từng từ trong input sequence. Từ nào chứa nhiều thông tin cần thiết cho việc decoding thì sẽ có giá trị weight lớn hơn và tổng các weights của tất cả các từ trong input sequence phải bằng 1. Giá trị Attention weights này được học thông qua quá trình huấn luyện với việc sử dụng input sequence và hidden state của decoding step trước (xem thêm tại đây). Mỗi Decoding step có một giá trị Attention vector riêng, do đó với một input sequence có chiều dài 'n' và output sequence có chiều dài 'm', ta phải thực hiện việc tính toán 'n * m' Attention weights. Điều này là có thể chấp nhận được trong các hệ thống Word-based do có số lượng từ không quá nhiều. Tuy nhiên với các hệ thống Character-based thì việc sử dụng Attention sẽ yêu cầu một tài nguyên xử lý lớn.

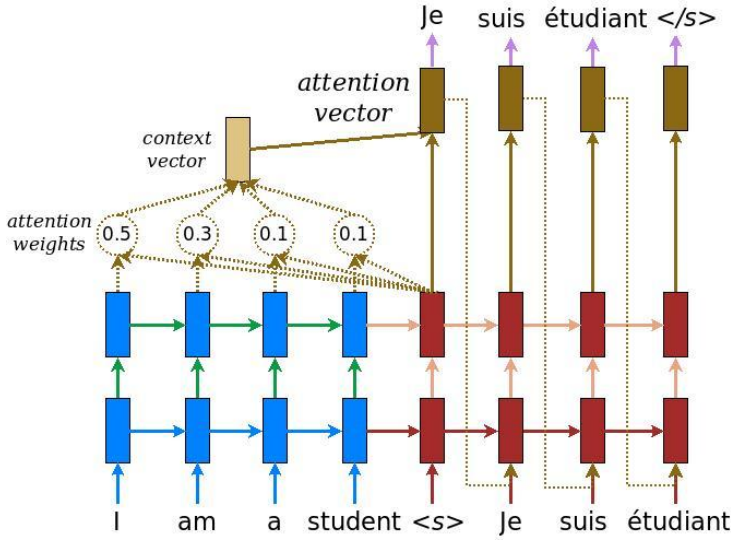


Figure 2.5: *Mô tả các mức độ tập trung được thiết lập bởi Attention weights*

2.4 Transformer [17]

Ý tưởng chính của bài viết này được lấy từ bài báo **attention is all you need** [11] giới thiệu về các kiểu attention model được sử dụng trong các tác vụ học máy. Trong bài báo cũng đưa ra một kiến trúc mới về Transformer hoàn toàn khác so với các kiến trúc RNN trước đây, mặc dù cả 2 đều thuộc lớp model seq2seq nhằm chuyển 1 câu văn input ở ngôn ngữ A sang 1 câu văn output ở ngôn ngữ B. Quá trình biến đổi (transforming) được dựa trên 2 phần encoder và decoder.

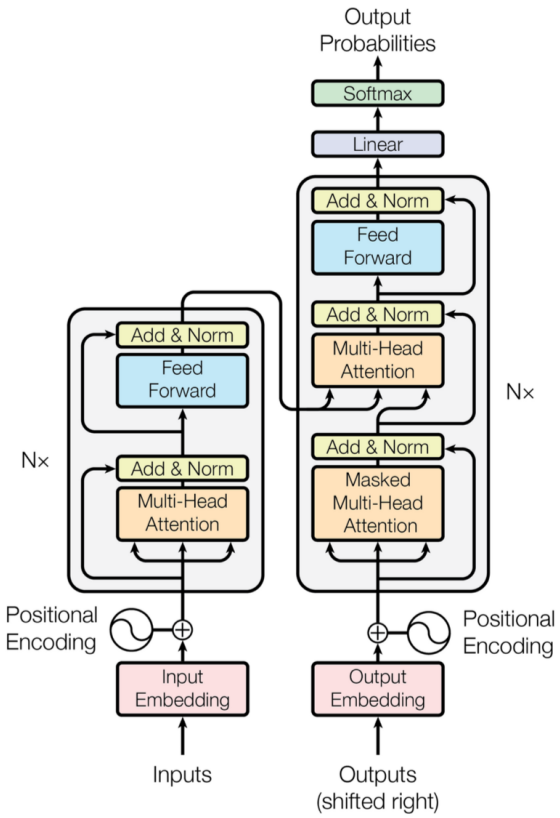


Figure 1: The Transformer - model architecture.

Figure 2.6: *Cấu trúc transformer*

Kiến trúc này gồm 2 phân encoder bên trái và decoder bên phải.

- Encoder: là tổng hợp xếp chồng lên nhau của 6 layers xác định. Mỗi layer bao gồm 2 layer con (sub-layer) trong nó. Sub-layer đầu tiên là multi-head self-attention mà lát nữa chúng ta sẽ tìm hiểu. Layer thứ 2 đơn thuần chỉ là các fully-connected feed-forward layer. Kiến trúc này có ý tưởng tương tự như mạng resnet trong CNN. Đầu ra của mỗi sub-layer là LayerNorm $(x + \text{Sublayer}(x))$ có số chiều là 512 theo như bài viết.
- Decoder: Decoder cũng là tổng hợp xếp chồng của 6 layers. Kiến trúc tương tự như các sub-layer ở Encoder ngoại trừ thêm 1 sub-layer thể hiện phân phối attention ở vị trí đầu tiên. Layer này không gì khác so với multi-head self-attention layer ngoại trừ được điều chỉnh để không đưa các từ trong tương lai vào attention. Tại bước thứ i của decoder chúng ta chỉ biết được các từ ở vị trí nhỏ hơn i nên việc điều chỉnh đảm bảo attention chỉ áp dụng cho những từ nhỏ hơn vị trí thứ i. Cơ chế residual cũng được áp dụng tương tự như trong Encoder.

2.5 Bidirectional Encoder Representation - BERT [17]

BERT [4] là model biểu diễn ngôn ngữ được google giới thiệu vào năm 2018. BERT được thiết kế để đào tạo ra các vector đại diện cho ngôn ngữ văn bản thông qua ngữ cảnh 2 chiều(trái và phải) của chúng. Kết quả là, vector đại diện được sinh ra từ mô hình BERT được tính chỉnh với các lớp đầu ra bổ sung đã tạo ra nhiều kiến trúc cải tiến đáng kể cho các nhiệm vụ xử lý ngôn ngữ tự nhiên như Question Answering, Language Inference,...mà không cần thay đổi quá nhiều từ các kiến trúc cũ.
Cơ chế attention của Transformer sẽ truyền toàn bộ các từ trong câu văn đồng thời vào mô hình một lúc mà không cần quan tâm đến chiều của câu. Do đó Transformer được xem như là huấn luyện hai chiều (bidirectional) mặc dù trên thực tế chính xác hơn chúng ta có thể nói rằng đó là huấn luyện không chiều (non-directional). Đặc điểm này cho phép mô hình học được bối cảnh của từ dựa trên toàn bộ các từ xung quanh nó bao gồm cả từ bên trái và từ bên phải. Trong đó, Bi-directional (hai chiều) là ngữ nghĩa của một từ không chỉ được biểu diễn bởi những từ liền trước mà còn được giải thích bởi toàn bộ các từ xung quanh. Luồng giải thích tuân theo đồng thời từ trái qua phải và từ phải qua trái cùng một lúc.

2.5.1 Fine-tuning model BERT

Một điểm đặc biệt ở BERT mà các model embedding trước đây chưa từng có đó là kết quả huấn luyện có thể fine-tuning được. Chúng ta sẽ thêm vào kiến trúc model một output layer để tùy biến theo tác vụ huấn luyện.

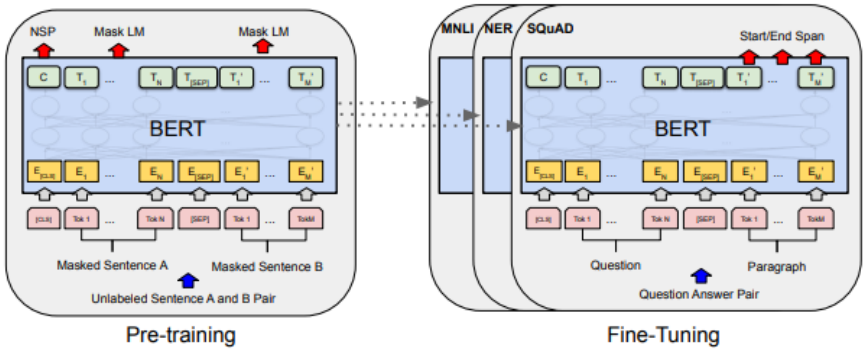


Figure 2.7: *Toàn bộ tiến trình pre-training và fine-tuning của BERT*

Một kiến trúc tương tự được sử dụng cho cả pretrain-model và fine-tuning model. Chúng ta sử dụng cùng một tham số pretrain để khởi tạo mô hình cho các tác vụ down stream khác nhau. Trong suốt quá trình fine-tuning thì toàn bộ các tham số của layers học chuyển giao sẽ được fine-tune. Đối với các tác vụ sử dụng input là một cặp sequence (pair-sequence) ví dụ như question and answering thì ta sẽ thêm token khởi tạo là [CLS] ở đầu câu, token [SEP] ở giữa để ngăn cách 2 câu.

Tiến trình áp dụng fine-tuning sẽ như sau:

- Bước 1: Embedding toàn bộ các token của cặp câu bằng các vector nhúng từ pretrain model. Các token embedding bao gồm cả 2 token là [CLS] và [SEP] để đánh dấu vị trí bắt đầu của câu hỏi và vị trí ngăn cách giữa 2 câu. 2 token này sẽ được dự báo ở output để xác định các phần Start/End Spand của câu output.
- Bước 2: Các embedding vector sau đó sẽ được truyền vào kiến trúc multi-head attention với nhiều block code (thường là 6,12 hoặc 24 blocks tùy theo kiến trúc BERT). Ta thu được một vector output ở encoder.
- Bước 3: Để dự báo phân phối xác suất cho từng vị trí từ ở decoder, ở mỗi time step chúng ta sẽ truyền vào decoder vector output của encoder và vector embedding input của decoder để tính encoder-decoder attention (cụ thể về encoder-decoder attention là gì các bạn xem lại mục 2.1.1). Sau đó projection qua liner layer và softmax để thu được phân phối xác suất cho output tương ứng ở time step t.
- Bước 4: Trong kết quả trả ra ở output của transformer ta sẽ cố định kết quả của câu Question sao cho trùng với câu Question ở input. Các vị trí còn lại sẽ là thành phần mở rộng Start/End Span tương ứng với câu trả lời tìm được từ câu input.

Lưu ý quá trình huấn luyện chúng ta sẽ fine-tune lại toàn bộ các tham số của model BERT đã cut off top linear layer và huấn luyện lại từ đầu các tham số của linear layer mà chúng ta thêm vào kiến trúc model BERT để customize lại phù hợp với bài toán.

2.5.2 Next Sentence Prediction (NSP)

Đây là một bài toán phân loại học có giám sát với 2 nhãn (hay còn gọi là phân loại nhị phân). Input đầu vào của mô hình là một cặp câu (pair-sequence) sao cho 50% câu thứ 2 được lựa chọn là câu tiếp theo của câu thứ nhất và 50% được lựa chọn một cách ngẫu nhiên từ bộ văn bản mà không có mối liên hệ gì với câu thứ nhất. Nhãn của mô hình sẽ tương ứng với IsNext khi cặp câu là liên tiếp hoặc NotNext nếu cặp câu không liên tiếp. Cũng tương tự như mô hình Question and Answering, chúng ta cần đánh dấu các vị trí đầu câu thứ nhất bằng token [CLS] và vị trí cuối các câu bằng token [SEP] . Các token này có tác dụng nhận biết các vị trí bắt đầu và kết thúc của từng câu thứ nhất và thứ hai.

Thông tin input được preprocessing trước khi đưa vào mô hình huấn luyện bao gồm:

- Ngữ nghĩa của từ (token embeddings): Thông qua các embedding vector cho từng từ. Các vector được khởi tạo từ pretrain model. Ngoài embedding biểu diễn từ của các từ trong câu, mô hình còn embedding thêm một số thông tin:
- Loại câu (segment embeddings): Gồm hai vector là E_A nếu từ thuộc câu thứ nhất và E_B nếu từ thuộc câu thứ hai.
- Vị trí của từ trong câu (position embedding): là các vector $E_0, \ldots, E_{10}$. Tương tự như positional embedding trong transformer.

vector input sẽ bằng tổng của cả ba thành phần embedding theo từ, câu và vị trí.

2.5.3 Các kiến trúc model BERT

Hiện tại có nhiều phiên bản khác nhau của model BERT. Các phiên bản đều dựa trên việc thay đổi kiến trúc của Transformer tập trung ở 3 tham số: L : số lượng các block sub-layers trong transformer, H : kích thước của embedding vector (hay còn gọi là hidden size), A : Số Iượng head trong multi-head layer, mối một head sẽ thực hiện một self-attention. Tên gọi của 2 kiến trúc bao gồm:

- BERT **BASE**$(L = 12, H = 768, A = 12)$: Tổng tham số 110 triệu.
- BERT **LARGE**$(L = 24, H = 1024, A = 16)$: Tổng tham số 340 triệu.

Như vậy ở kiến trúc BERT Large chúng ta tăng gấp đôi số layer, tăng kích thước hidden size của embedding vector gấp 1.33 lần và tăng số lượng head trong multi-head layer gấp 1.33 lần.

2.5.4 PhoBERT [12]

PhoBERT là một pre-trained được huấn luyện monolingual language, tức là chỉ huấn luyện dành riêng cho tiếng Việt. Việc huấn luyện dựa trên kiến trúc và cách tiếp cận giống RoBERTa [13] của Facebook được Facebook giới thiệu giữa năm 2019. PhoBERT có 2 phiên bản:

- phoBERT-base: số lượng param là 135M, chiều dài tối đa là 256.
- phoBERT-large: số lượng param là 370M, chiều dài tối đa là 256.

2.6 BART [15]

BART [14] là mô hình được giới thiệu bởi Facebook AI, một mô hình pretrained mới kết hợp ưu điểm của BERT và GPT. Sức mạnh của BERT nằm ở việc nắm bắt ngữ cảnh hai chiều, trong khi đó GPT có khả năng tự hồi quy. Với sự ra đời của BART, các nhiệm vụ sinh và đọc hiểu văn bản có thể được thực hiện với cùng một mô hình.

BART là một autoencoder khử nhiễu trên kiến trúc sequence-to-sequence, có thể được áp dụng trong đa dạng các nhiệm vụ khác nhau. Nó sử dụng kiến trúc transformers chuẩn cho bài toán dịch máy. Việc huấn luyện BART bao gồm việc tạo nhiễu trong văn bản với một hàm tùy ý và sử dụng mô hình để tái cấu trúc lại văn bản ban đầu. Ưu điểm chính của cách thức này là mô hình trở nên linh hoạt với văn bản đầu vào và tái tạo lại văn bản một cách hiệu quả.

BART cho thấy hiệu quả vượt trội trong cả nhiệm vụ sinh lẫn đọc hiểu văn bản. Cụ thể, BART có hiệu quả sánh ngang RoBERTa trên GLUE và SQuAD và đạt SOTA trong các nhiệm vụ về đối thoại trừu tượng, trả lời câu hỏi và tóm tắt.

2.6.1 BARTpho [18]

BARTpho là một cải tiến mới từ phoBERT. Hiện nay, BARTpho đã có 2 phiên bản, BARTpho-syllable và BARTpho-word, đó là những mô hình seq2seq dành riêng cho tiếng việt. BARTpho sử dụng kiến trúc "Bart Large" (với chiều dài tối đa là 1024) và pre-training từ mô hình autoencoder BART. Do đó đặc biệt thích hợp với các nhiệm vụ NLP.

- BARTpho-syllabel: số lượng param 396M, chiều dài tối đa 1024.
- BARTpho-syllabel: số lượng param 420M, chiều dài tối đa 1024.

2.7 Các chỉ số đánh giá

- True Positive (TP): sô lượng điểm của lớp positive được phân loại đúng là positive.
- True Negative (TN): số lượng điểm của lớp negative được phân loại đúng là negative.
- False Positive (FP): số lượng điểm của lớp negative bị phân loại nhầm thành positive.
- False Negative (FN): số lượng điểm của lớp positive bị phân loại nhầm thành negative
- True positive rate (TPR), false negative rate (FNR), false positive rate (FPR), true negative rate (TNR) sẽ được mô tả trong bảng sau.

	Predicted as Positive	Predicted as Negative
Actual: Positive	TPR = TP/(TP + FN)	FNR = FN/(TP + FN)
Actual: Negative	FPR = FP/(FP + TN)	TNR = TN/(FP + TN)

2.7.1 Accuracy

Khi xây dựng mô hình phân loại chúng ta sẽ muốn biết một cách khái quát tỷ lệ các trường hợp được dự báo đúng trên tổng số các trường hợp là bao nhiêu. Tỷ lệ đó được gọi là độ chính xác. Độ chính xác giúp ta đánh giá hiệu quả dự báo của mô hình trên một bộ dữ liệu.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

2.7.2 F1 - score

F1 Score là trung bình điều hòa giữa precision và recall. Do đó nó đại diện hơn trong việc đánh gía độ chính xác trên đồng thời precision và recall.

$$F_1 = 2\frac{1}{\frac{1}{\text{precision}} + \frac{1}{\text{recall}}} = 2\frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Khi kích thước các lớp dữ liệu là chênh lệch (imbalanced data hay skew data), precision và recall thường được sử dụng

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \text{ Recall } = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

F_1-score có giá trị nằm trong nửa khoảng $(0, 1]$. F_1 càng cao, bộ phân lớp càng tốt. Khi cả recall và precision đều bằng 1 (tốt nhất có thể), $F_1 = 1$. Khi cả recall và precision đều thấp, ví dụ bằng 0.1, $F_1 = 0.1$. Dưới đây là một vài ví dụ về F_1

precision	recall	F_1
1	1	1
0.1	0.1	0.1
0.5	0.5	0.5
1	0.1	0.182
0.3	0.8	0.36

Như vậy, một bộ phân lớp với precision = recall = 0.5 tốt hơn một bộ phân lớp khác với precision = 0.3, recall = 0.8 theo cách đo này.

Chương 3

Phương pháp tiếp cận

3.1 Mô tả phương pháp

Hai nhiệm vụ chính là Aspect Extraction (AE) và Aspect Polarity Classification (APC) nên ý tưởng ban đầu cho các mô hình là từ dữ liệu thu thập được chuyển qua giai đoạn xử lý dữ liệu, mã hóa, qua một lớp embedding và phân loại theo các nhãn của AE và APC.

Figure 3.1: *Quy trình triển khai các mô hình*

3.2 Xây dựng một số mô hình

1. **Mô hình cho nhiệm vụ AE**:
   Hai mô hình được sử dụng để trích xuất aspect, đầu vào là một câu và sử dụng **phoBert** [12] và **BARTpho** [18] để embedding, kết quả sau khi embedding sẽ lấy phần **"last_hidden_state"** (là đầu ra của lớp cuối cùng của base-model) được đưa qua một lớp linear và trả về xác suất của ba label aspect cho từng token. Hàm cho Loss cho AE: cross-entropy loss

$$\mathcal{L}_{AE} = \sum_{1}^{N} \sum_{1}^{k} \widehat{t_i} \log t_i$$

   trong đó, N là số lượng nhãn aspect và k là tổng số lượng token trong mỗi câu đầu vào, với t_i là nhãn aspect của mỗi token và $\widehat{t_i}$ là dự đoán aspect cho từng token.

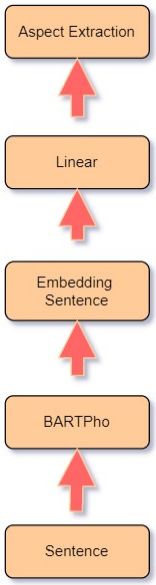


Figure 3.2: *Model A*

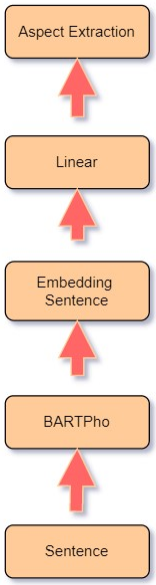


Figure 3.3: *Model F*

2. **Mô hình cho nhiệm vụ APC**:

   Cách tiếp cận 1

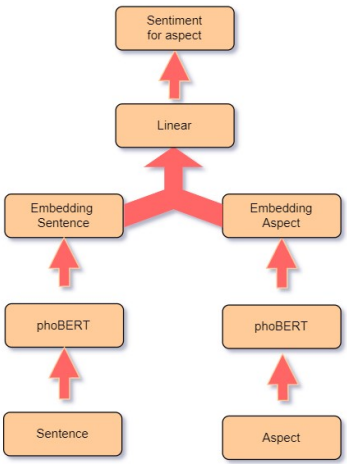


Figure 3.4: *Model B*

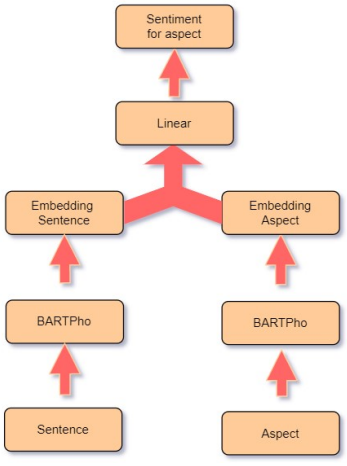


Figure 3.5: *Model G*

Hai mô hình trên được dùng để phân loại sentiment cho các aspect đã được định sẵn. Đầu vào là 1 câu và 1 aspect đã trích xuất sẵn, dựa vào **phoBert** [12] và **BARTpho** [18] để embedding cho câu và aspect, kết quả sau khi đã embedding sẽ lấy phần **"last_hidden_state"** (là đầu ra lớp cuối cùng của base-model) của mỗi loại để kết hợp lại bằng hàm concat và đưa qua một lớp linear nhận lại đầu ra với xác suất của 3 label sentiment (negative, positive, neutral).

Cách tiếp cận 2

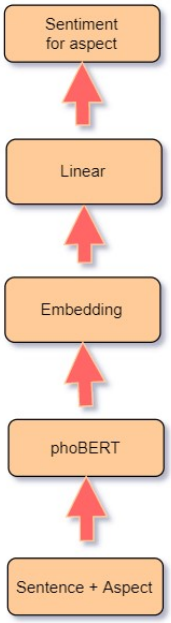


Figure 3.6: *Model C*

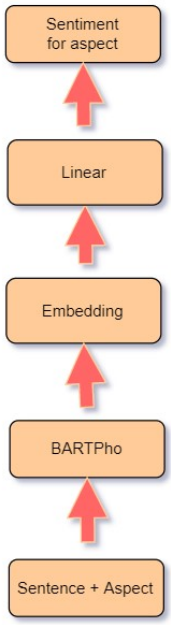


Figure 3.7: *Model H*

Hai mô hình trên được dùng để phân loại sentiment cho các aspect đã được định sẵn. Đầu vào là 1 câu nối liền với 1 aspect đã trích xuất sẵn, dựa vào **phoBert** [12] và **BARTpho** [18] để embedding, kết quả sau khi đã embedding sẽ lấy phần **"last_hidden_state"** (là đầu ra của lớp cuối cùng của base-model) đưa qua một lớp linear và nhận lại đầu ra với xác suất của 3 label sentiment (negative, positive, neutral).

Hàm cho Loss cho APC: cross-entropy loss

$$\mathcal{L}_{APC} = \sum_{1}^{C} \widehat{y}_i \log y_i$$

trong đó, C là số lượng nhãn sentiment, với y_i là nhãn sentiment của aspect và $\widehat{y}_i$ là dự đoán sentiment cho aspect.

3. **Mô hình multi task**:

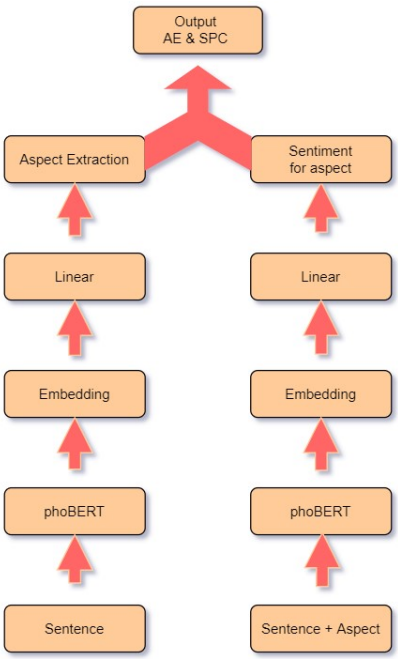


Figure 3.8: *Model D*

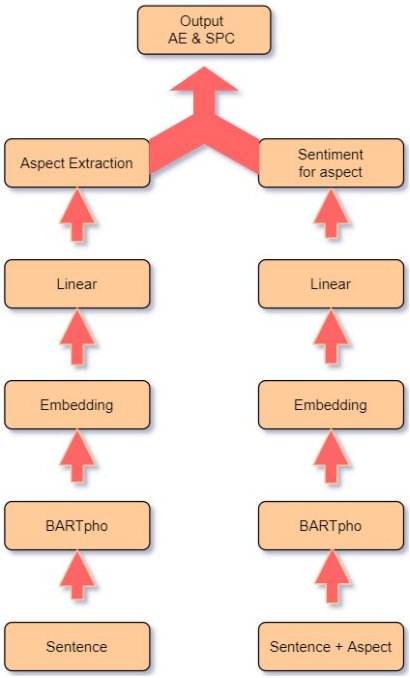


Figure 3.9: *Model I*

Mô hình này kết hợp từ 2 nhiệm vụ AE và SPC (**theo cách tiếp cận 2**) và được huấn luyện cùng lúc 2 nhiệm vụ song song, đầu ra bao gồm cả 2 loại là phân loại token cho nhiệm vụ AE và phân loại cảm xúc cho aspect cho nhiệm vụ SPC.

Hàm loss cho multi-task: Cross-entropy loss

$$\mathcal{L} = \mathcal{L}_{AE} + \mathcal{L}_{APC}$$

4. **Mô hình multi task kết hợp Attention**:

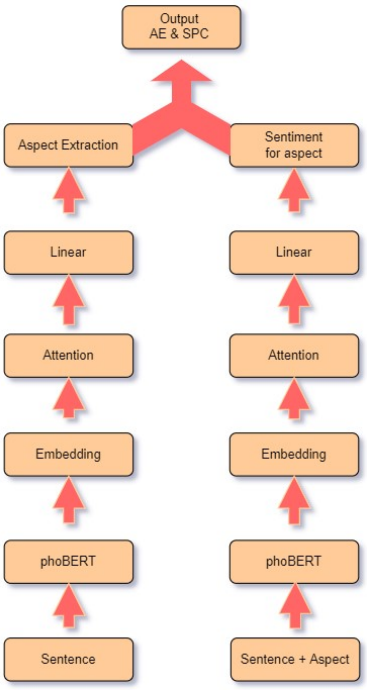


Figure 3.10: *Model E*

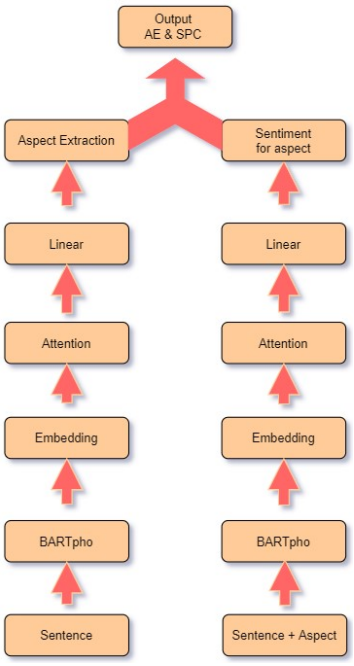


Figure 3.11: *Model J*

Thêm cơ chế Attention để mô hình có thể tập trung các từ quan trọng. Vì aspect có 2 yếu tố là B-asp và I-asp, để tránh các trường hợp các aspect bị sai lệch quá lớn với những từ không phải aspect.

$$Attention(Q, K, V, adj\_matrix) = softmax\left(\frac{Q.K^T}{\sqrt{d_k}} + adj\_matrix\right).V$$

Trong đó, Q, K, V được lấy từ kết quả lớp cuối cùng của embedding (**"last_hidden_state"**), d_k là só chiều của một biến độc lập và adj_matrix (adjacency matrix) là ma trận lấy từ các cầu nối của các parsing trong câu.

Hàm loss: Cross-entropy loss

$$\mathcal{L} = \mathcal{L}_{AE} + \mathcal{L}_{APC}$$

Với hàm loss của AE tôi có thiết lập thêm trọng số để tránh việc bị lệch số lượng label.

Chương 4

Thực nghiệm

4.1 Mô tả dữ liệu

Dữ liệu được lấy từ bình luận của khách hàng trên trang web thương mại điện tử và được dán nhãn thủ công.

- Bình luận/đánh giá của khách hàng.
- aspect labels: B-asp, I-asp, O.
- sentiment labels: negative (0), positive (1), neutral (2), the rest (-1).

Số lượng thu thập dữ liệu là 10000 câu.

Cách gán nhãn thủ công:

Sản	phẩm	này	rất	tốt	giá	cả	phù	hợp	giao	hàng	chậm
B-asp	I-asp	O	O	O	B-asp	O	O	O	B-asp	I-asp	O
1	1	-1	-1	-1	2	-1	-1	-1	0	0	-1

Mỗi chuỗi được gán 2 loại nhãn, nhãn đầu tiên cho biết token đó có phải là một aspect hay không và gán nhãn theo quy tắc IOB. Trong đó, B là bắt đầu, I là phần nối tiếp, O là phần còn lại. Nhẫn thứ hai là đánh dấu sentiment của các token là aspect.

4.2 Tiền xử lý dữ liệu

Một trong những bước quan trọng trong NLP, cụ thể hơn là dữ liệu văn bản được thu thập từ các trang web thương mại điện tử. Đó là lý do tại sao chúng ta cần loại bỏ những từ không chính thống và không phù hợp với chuẩn mực thông thường của tiếng Việt. Sử dụng một số bộ tiền xử lý cơ bản như:

- Chuẩn hóa chữ thường.
- Tách từ.
- Xử lý các trường hợp dấu câu lặp lại (Ví dụ: Ngon !!!!! $\Longrightarrow$ Ngon !).
- Loại bỏ các câu không có aspect.
- Mã hóa

Trong quá trình tiền xử lý, để dữ liệu được làm sạch và chuyển đổi sang định dạng phù hợp, quá trình chuyển đổi văn bản đã cho thành một chuỗi mã biểu diễn được gọi là tokenization. Tokenization giúp hình thành dạng vector để máy có thể dễ dàng học được dữ liệu văn bản thông qua vector, đây là một bước tiền xử lý quan trọng. Quá trình chuyển đổi mã biểu diễn của các từ thành định dạng vector được gọi là word embedding. Có các từ gần như giống nhau về mặt hình thức, nhưng có sự khác biệt về nghĩa. Để làm cho một cỗ máy hiểu được sự khác biệt về ý nghĩa, người ta sử dụng tính năng nhúng từ để chuyển văn bản thành một thứ nguyên khác.

4.3 Phân tích dữ liệu

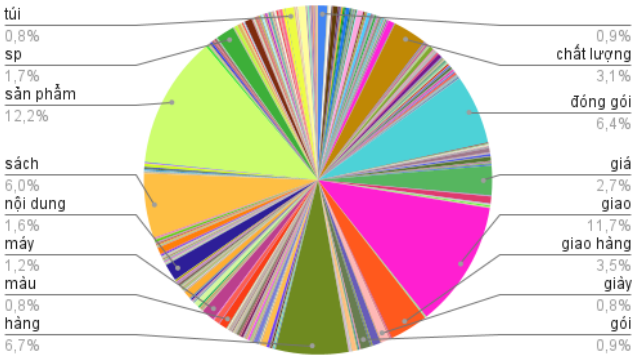


Figure 4.1: *Hình trích lọc các aspect được từ dữ liệu*

Các khía cạnh được trích lọc từ dữ liệu, sau khi gán nhãn thủ công. Các khía cạnh đa dạng và số lượng của mỗi loại thì không đều, có các khía cạnh khách hàng sử dụng lại rất nhiều lần và cũng có các khía cạnh chỉ xuất hiện vài lần.

TRAIN SET

		Aspect			Sentiment		
Menu	**Số câu**	**B-asp**	**I-asp**	**O**	**Neg**	**Pos**	**Neu**
Bách Hóa Online	303	5748	913	6136	649	5043	279
Nhà Sách Tiki	1519	37169	7709	42916	5391	31926	2870
Đồ Chơi - Mẹ Và Bé	601	12462	1394	13325	2229	10579	390
Làm Đẹp - Sức Khỏe	591	14495	4041	15238	2277	12527	1481
Thời Trang	573	11488	1869	11616	2381	9319	473
Nhà Cửa - Đời Sống	917	19995	6106	20413	5376	16268	2522
Thể Thao - Dã Ngoại	240	5130	919	5228	761	4213	327
Điện Tử - Điện Lạnh	182	5304	1085	7169	1095	4092	525
Máy Ảnh	114	2404	187	2627	530	1887	85
Ô Tô - Xe Máy - Xe Đạp	149	2822	349	2864	825	1782	262
Thiết Bị Số - Phụ Kiện Số	867	20615	4873	22008	4174	15972	2471
Điện Gia Dụng	263	7255	1169	7476	1071	5989	921
Laptop - Linh kiện	174	5169	1734	5594	1124	3917	428
Voucher - Dịch vụ	6	143	17	143	17	126	0
Hàng Quốc Tế	149	3133	881	3299	264	2580	617
Điện Thoại	9	270	104	270	83	126	144
Tổng	**6657**	**153602**	**33350**	**166322**	**28247**	**126346**	**13795**

VALID SET

		Aspect			Sentiment		
Menu	**Số câu**	**B-asp**	**I-asp**	**O**	**Neg**	**Pos**	**Neu**
Bách Hóa Online	86	1599	163	1781	180	1353	83
Nhà Sách Tiki	363	8830	1293	10346	1152	7667	549
Đồ Chơi - Mẹ Và Bé	154	3416	310	3533	499	2960	30
Làm Đẹp - Sức Khỏe	161	3975	610	4007	519	3394	410
Thời Trang	158	3151	504	3168	643	2609	179
Nhà Cửa - Đời Sống	240	5191	1873	5244	1322	4064	1002
Thể Thao - Dã Ngoại	62	1165	150	1291	53	1075	59
Điện Tử - Điện Lạnh	36	1419	167	1543	307	1074	51
Máy Ảnh	18	301	12	397	61	242	48
Ô Tô - Xe Máy - Xe Đạp	41	1014	64	1233	158	844	50
Thiết Bị Số - Phụ Kiện Số	207	4750	1220	5157	762	3935	490
Điện Gia Dụng	58	1644	436	1666	659	1010	391
Laptop - Linh kiện	42	933	185	991	196	689	0
Voucher - Dịch vụ	2	48	0	48	0	48	0
Hàng Quốc Tế	36	711	318	711	20	582	223
Điện Thoại	2	49	0	49	27	22	0
Tổng	**1666**	**38196**	**7305**	**41165**	**6558**	**6558**	**31568**

Dữ liệu sau khi đã xử lý thì được chia thành 2 tập train, valid để huấn luyện mô hình. Có thể thấy dữ liệu bị lệch nặng, ví dụ như mục Voucher - Dịch vụ trong cả TRAIN SET và VALID SET hầu như không có sự xuất hiện của neutral, trong khi postive thì chiếm đa phần.

4.4 Triển khai

Tôi thực hiện các thử nghiệm mô hình trên bằng Pytorch và sử dụng **phoBERT**, **BARTpho** làm base-model để embedding đầu vào. Các mô hình trên đều dùng Adam để tối ưu các tham số của mô hình và hàm loss đều sử dụng cross-entropy. Cuối cùng tôi đánh giá các mô hình dựa trên thang đo Accuracy và F_1-score (trung bình của mỗi class).

4.5 Kết quả

		acc_asp	**acc_senti**	F_1**-asp**	F_1**-senti**
phoBERT	Mô hình A	95.26	-	53.72	-
	Mô hình B	-	80.56	-	35.23
	Mô hình C	-	87.21	-	41.40
	Mô hình D	94.26	86.99	44.99	40.64
	Mô hình E	84.30	86.62	36.61	40.80
BARTpho	Mô hình F	94.99	-	61.29	-
	Mô hình G	-	85.74	-	40.98
	Mô hình H	-	88.18	-	49.95
	Mô hình I	95.36	90.68	**61.34**	**55.54**
	Mô hình J	90.22	87.48	43.32	41.90

Tổng quan thì có thể thấy BARTpho có sự vượt trội hơn hẳn với phoBERT. Bên cạnh đó, với nhiệm vụ APC thì cách tiếp cận thứ 2 tốt hơn so cách tiếp cận thứ nhất có thể so sánh từ mô hình B,C và G,H. Còn mô hình kết hợp thêm attention bên ngoài chưa được tốt như mong đợi.

Chương 5

Kết luận và hướng phát triển

Đây là một bài toán rất hay để có thể ứng dụng cho tiếng việt, thật sự là một thử thách. Mô hình BARTpho đã cho thấy mức độ hiệu quả hơn hẳn so với mô hình phoBERT nhưng tổng quan lại vẫn chưa đạt được kết quả vượt trội, do các yếu tố về số lượng dữ liệu và còn các trường hợp dữ liệu chưa được xử lý sạch sẽ và các thiếu sót trong công tác gán nhãn thủ công chưa thể xử lý hết.

Để phát triển thêm thì có thể mở rộng bộ nhớ để tránh việc bình luận bị ảnh hưởng từ các bình luận của nhiều người khác về một món hàng và thu thập nhiều dữ liệu hơn, đa dạng hơn và thử nghiệm với các mô hình khác nhằm cải thiện hiệu suất cho hai nhiệm vụ chính là AE và APC. Đồng thời có thể tìm hiểu và áp dụng thêm graph convolution networks cho bài toán, đi sâu hơn về cú pháp và cấu trúc câu.

Tài liệu tham khảo

[1] Heng Yang,Biqing Zeng,JianHao Yang,Youwei Song,Ruyang Xu, "A Multi-task Learning Model for Chinese-oriented Aspect Polarity Classification and Aspect Term Extraction", arXiv:1912.07976v3 [cs.CL] 12 Feb 2020.

[2] Natural-language-processing-NLP, https://www.techtarget.com/searchenterpriseai/definition/natural-language-processing-NLP

[3] Named entity recognition https://viblo.asia/p/seri-nlp-nhan-dang-thuc-the-ner-phan-1-Ljy5VyWzlra

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", arXiv:1810.04805v2 [cs.CL] 24 May 2019.

[5] Word2vec, https://machinelearningcoban.com/tabml_book/ch_embedding/word2vec

[6] RNN, https://phamdinhkhanh.github.io/2019/04/22/Ly_thuyet_ve_mang_LSTM

[7] AttentionLayer, https://phamdinhkhanh.github.io/2019/06/18/AttentionLayer

[8] Sequence to sequence model, http://itechseeker.com/tutorials/nlp-with-deep-learning/ly-thuyet-chung/sequence-to-sequence-model

[9] Autoencoder, https://viblo.asia/p/tim-hieu-ve-autoencoder-oOVlYvJv58W

[10] attention-mechanism,http://itechseeker.com/tutorials/nlp-with-deep-learning/ly-thuyet-chung/attention-mechanism

[11] Ashish Vaswani, Noam Shazeer,Niki Parmar, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, "Attention Is All You Need", https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[12] Nguyen Tuan Nguyen, Findings 2020 "PhoBERT: Pre-trained language models for Vietnamese", https://aclanthology.org/2020.findings-emnlp.92.pdf

[13] Yinhan Liu, Myle Ott, Naman Goyal, RoBERTa: A Robustly Optimized BERT Pretraining Approach, arXiv:1907.11692

[14] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, Luke Zettlemoyer, "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension", arXiv:1910.13461v1 [cs.CL] 29 Oct 2019

[15] BART: Sự kết hợp giữa BERT và GPT, trituenhantao.io/kien-thuc/bart-su-ket-hop-giua-bert-va-gpt

[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin, "Attention Is All You Need", arXiv:1706.03762v5 [cs.CL] 6 Dec 2017

[17] Pham Dinh Khanh, "BERT model", https://phamdinhkhanh.github.io/2020/05/23/BERTModel.html

[18] Nguyen Luong Tran, Duong Minh Le, Dat Quoc Nguyen, "BARTpho: Pre-trained Sequence-to-Sequence Models for Vietnamese", arXiv:2109.09701v2 [cs.CL] 2 Jan 2022.

[19] Wang, Y., Huang, M., Zhu, X., Zhao, L., 2016. Attention-based lstm for aspect-level sentiment classification, in: Proceedings of the 2016 conference on empirical methods in natural language processing, pp. 606–615. doi:10.18653/v1/d16-1058.

[20] Peng, H., Ma, Y., Li, Y., Cambria, E., 2018. Learning multi-grained aspect target sequence for chinese sentiment analysis. Knowledge-Based Systems 148, 167–176. doi:10.1016/j.knosys.2018.02.034.

[21] Song, Y., Wang, J., Jiang, T., Liu, Z., Rao, Y., 2019. Attentional encoder network for targeted sentiment classification. arXiv preprint arXiv:1902.09314 URL: https://arxiv.org/abs/1902.09314.

[22] Rietzler, A., Stabinger, S., Opitz, P., Engl, S., 2019. Adapt or get left behind: Domain adaptation through bert language model finetuning for aspect-target sentiment classification. arXiv preprint arXiv:1908.11860 URL: https://arxiv.org/abs/1908.11860.